Traffic Light Project

Tasheena Narraidoo and Caroline MacGillivray

09/25/17

1. Description

Introduction

In the Traffic Light Project, turtles are cars traveling across a bridge. The bridge only allows one-way traffic. In the model, turtles are cars traveling on a road. While the road itself is two-way, in the middle of the road there is a bridge which only allows one-way traffic. The bridge directs traffic from the westbound and eastbound lanes with the use of a stoplight.

Goal

The goal is to optimize the conditions of the traffic light so as to maximize efficiency. We define efficiency by two measures: wait time for the cars and the number of cars stopped. With the frequency and speed of cars fixed as "the rules of the road," we have to adjust the traffic light itself to achieve our goal.

Our model

We have removed the option where the user can manually input how long the light will remain green. This is because we wanted a set of rules that would automatically assess when the lights change.

In our model, when the westbound light is green and eastbound light is red, if the number of eastbound stopped cars is greater than the number of westbound stopped cars with a buffer that the user can adjust, then the westbound light turns red. We then have both red westbound and eastbound lights.

Similarly, when eastbound light is green and westbound light is red, if the number of westbound stopped cars is greater than the number of eastbound stopped cars with a buffer that the user can adjust, the eastbound light turns red. We then again have both red westbound and eastbound lights.

When both the eastbound and westbound lights turn red, our model then waits for the bridge to clear before changing colors and carrying on with the default settings. We created a buffer slider to take into account the speed cars are traveling at. In this simple model, we have fixed the speed of cars but we created the buffer slider to analyze the traffic. For high speed traffic, we thought it would be better to have a higher buffer as compared to a slower speed traffic. Otherwise a small buffer will make the lights change too fast.

2.ODD

We have followed the ODD (Overview Design Details) protocol designed by Grimm et al. to describe our model.

Purpose

The goal is to adjust the settings of the traffic light so as to minimize the number of stopped cars at the intersection at any given moment, as well as the cars' wait times. With the frequency, speed limit, and acceleration selected, we have to adjust the traffic light to achieve our goal.

Parameter and Variable Description

Note that in our model, 1 patch = 20 feet. We have fixed the speed of the cars to be speed limit to 2.2 patches/tick (about 30mph), and the maximum acceleration of the cars to be .3 ticks/seconds^2 (about 5.7 ft/seconds^2).

Parameter Description

Parameters	Description	Scale/Metrics
FREQ-EAST	select how often new eastbound cars are added.	probability of a car being added on each tick
FREQ-WEST	select how often new westbound cars are added.	probability of a car being added on each tick
BUFFER-TURTLES	fixes the difference between the waiting length.	1 unit = 1 car size

Variable Description

Variables	Description	Scale/Metrics
waiting-eastbound	number of stopped cars from eastbound	number of cars per tick
waiting-westbound	number of stopped cars from westbound	number of cars per tick
avg-west-wait	average wait time of stopped cars from westbound	number of ticks
avg-east-wait	average wait time of stopped cars from eastbound	number of ticks
waiting overall	total number of stopped cars at every tick	number of cars per tick
avg wait	average wait time of stopped cars over all cars	number of ticks
CLOCK	shows how many ticks have elapsed.	1 tick = 1 second

Process overview

The road is initialized with the "setup" command to draw the road. Upon the "go" command, the model proceeds one second at a time. One side will have the red light and another the green; when the number of cars on the stopped side exceeds the number of cars on the other side plus the pre-assigned buffer, the light will switch on its own.

As previously noted, we have removed the option where the user can manually input how long the light will remain green. This is because we wanted a set of rules that would automatically assess when the lights change. In our model, when the westbound light is green and eastbound light is red, if the number of eastbound stopped cars is greater than the number of westbound stopped cars with a buffer that the user can adjust, then the westbound light turns red. We then have both red westbound and eastbound lights.

Similarly, when eastbound light is green and westbound light is red, if the number of westbound stopped cars is greater than the number of eastbound stopped cars with a buffer that the user can adjust, the eastbound light turns red. We then again have both red westbound and eastbound lights.

When both the eastbound and westbound lights turn red, our model then waits for the bridge to clear before changing colors and carrying on with the default settings. In this simple model, we have fixed the speed of cars but we created the buffer slider to analyze the traffic.

Design

Basic principles The metrics (as given above) are fixed. The speed limit and maximum acceleration are also fixed. Unlike the Game of Life, something significant doesn't necessarily happen at each step of the model (i.e. the road doesn't change at every step.) The model is allowed to run continuously; although for all the tests discussed in this report, the time limit is 1000 ticks=1000 seconds=about 17 minutes.

Sensing

The traffic light can sense the number of cars waiting in each direction and is aware of the number of buffer turtles needed to switch the light. The turtles can sense when the light is red and when the light is green. They can also sense when a car is in front of them so that they can slow down.

Observation

We used two measures to gauge the effect of the buffer turtle parameter. At the end of a 1000 tick run, it measures the average number of cars waiting in any given tick, given by cumulative-stopped/1000. The model also measures the average waiting time among all cars, given by avg-wait-overall/max list 1 ticks (this avoids dividing by zero).

Details

Initialization

When a user initializes the road with the "setup" command, NetLogo sets cumulative stopped and average wait overall to 0, speed limit to 2.2 patches/tick (about 30mph), and max-accel to .3 ticks/seconds^2 (about 5.7 ft/seconds^2)

Everything else in the setup procedure was part of the original model.

Inputs

There are no inputs for the traffic light model.

Submodels

Move-cars: This is the same as in the original model.

Make-new-cars: The model is capable of creating new cars while it runs. If the number of cars is less than 100 (a number given in the original model and which we did not change), it will create one car.

Update the stop light: Except for our rule, the stop light runs pretty much as in the default model. The light knows to let the bridge clear before switching to green.

While designing our model, the original base model was modified to allow for the speed limit and acceleration values to be 2.2 and 0.3, whereas the original model took integers for those values. As opposed to just making the sliders work in smaller units, the updates also readjusted the road based on the coordinates rather than patch color.

To update the stop light: The light switches when the number of cars waiting at the red light plus the number of buffer turtles is greater than the number of cars in the lane with the green light. The light knows to let the bridge clear before switching. Everything else is as in the default model.

- On setup, the westbound light is red while that of eastbound is green.

3. Graphs of simulation

Below are some screen shots of our model under three scenarios. From the netlogo interface of our model, we can see the parameters on the left (freq-eastbound, freq-westbound, buffer-turtles) and some of our variables (waiting overall, waiting-eastbound, waiting-westbound) on the right. We also have a plot that shows the trends of the number of stopped cars from each lane and the overall result.

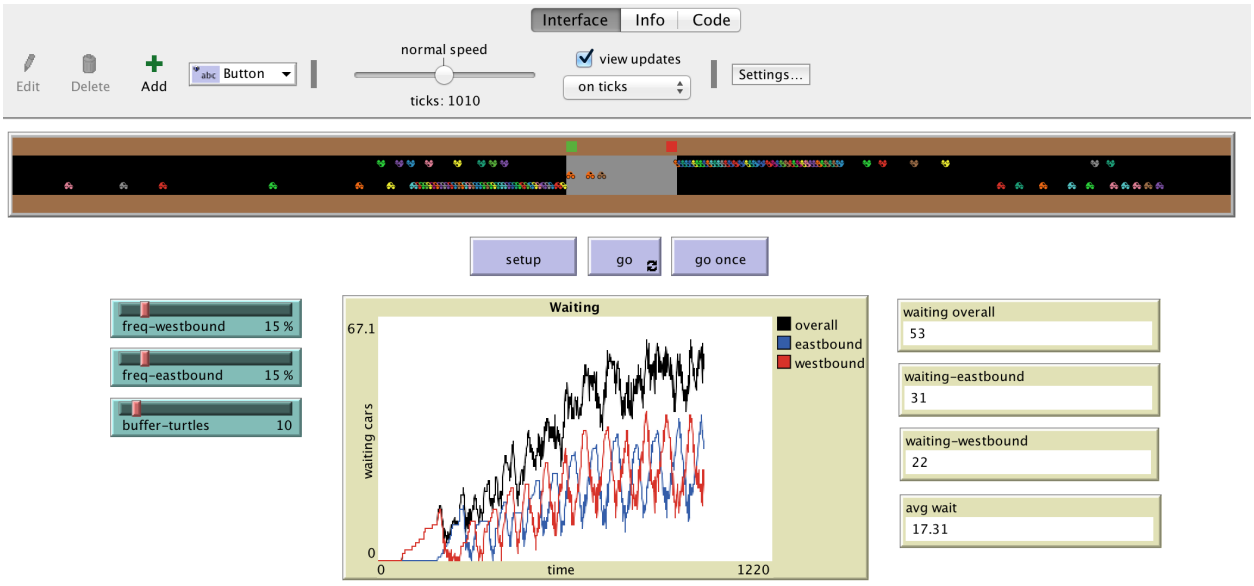

Figure 1: Light Traffic

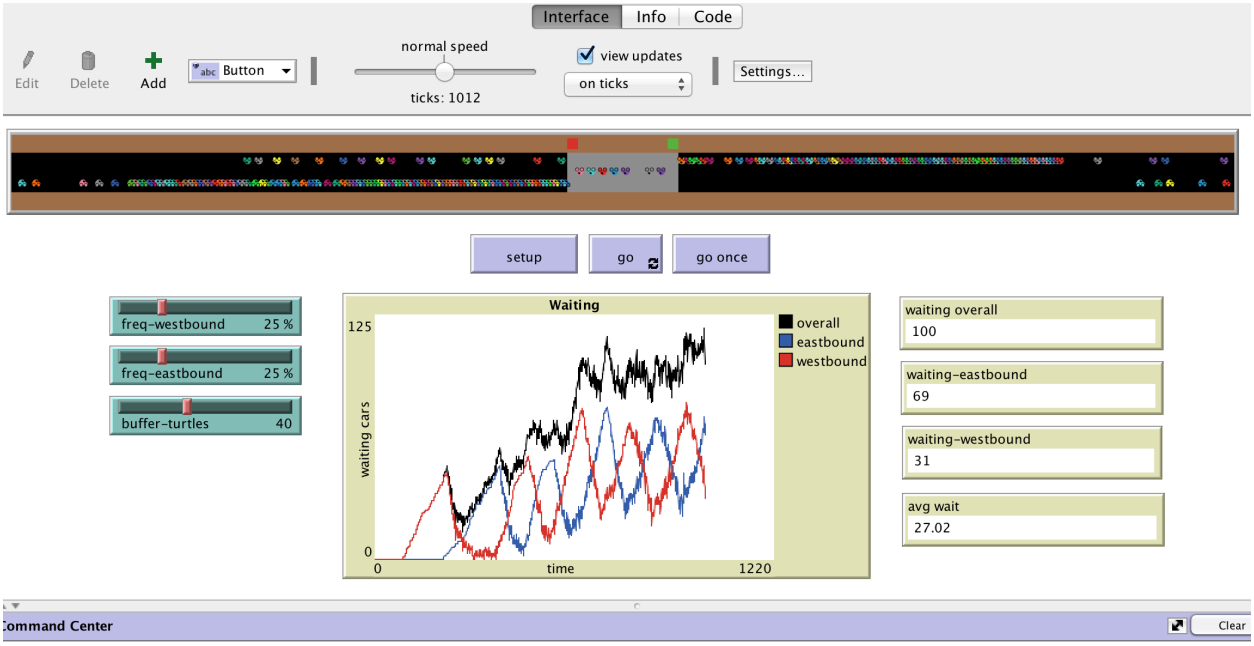

Figure 2: Heavy Traffic

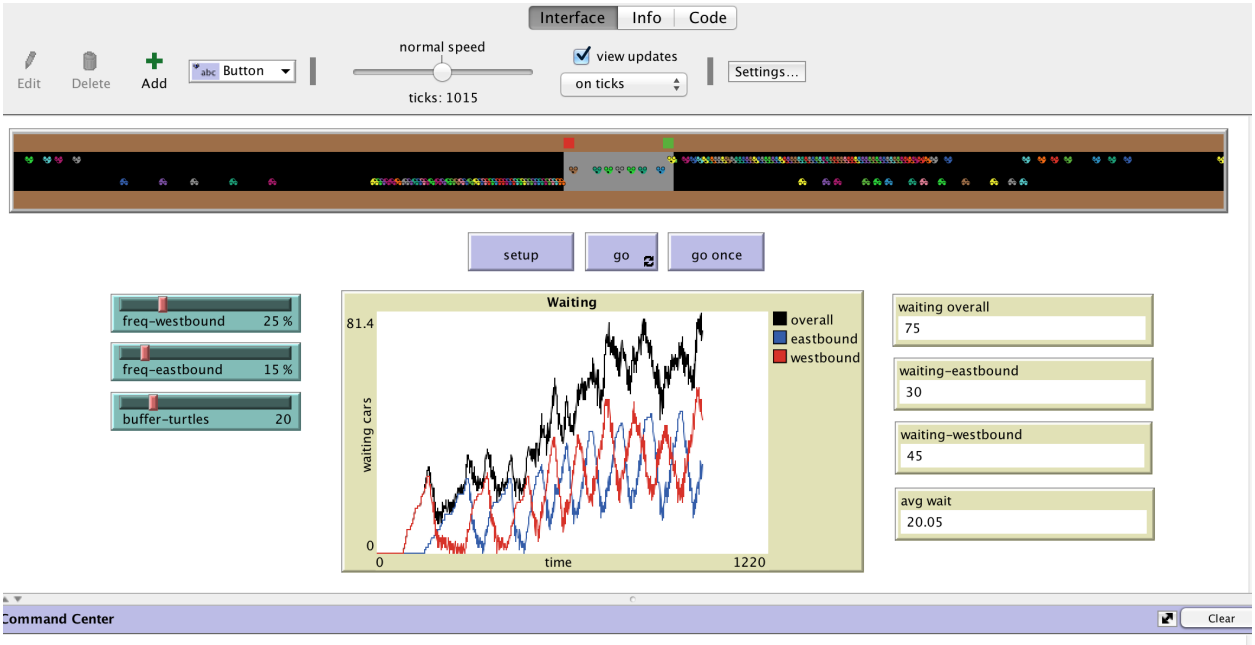

Figure 3: One-sided Heavy Traffic

4. Graphs summarizing results

Our results can be summarized in the following graphs.

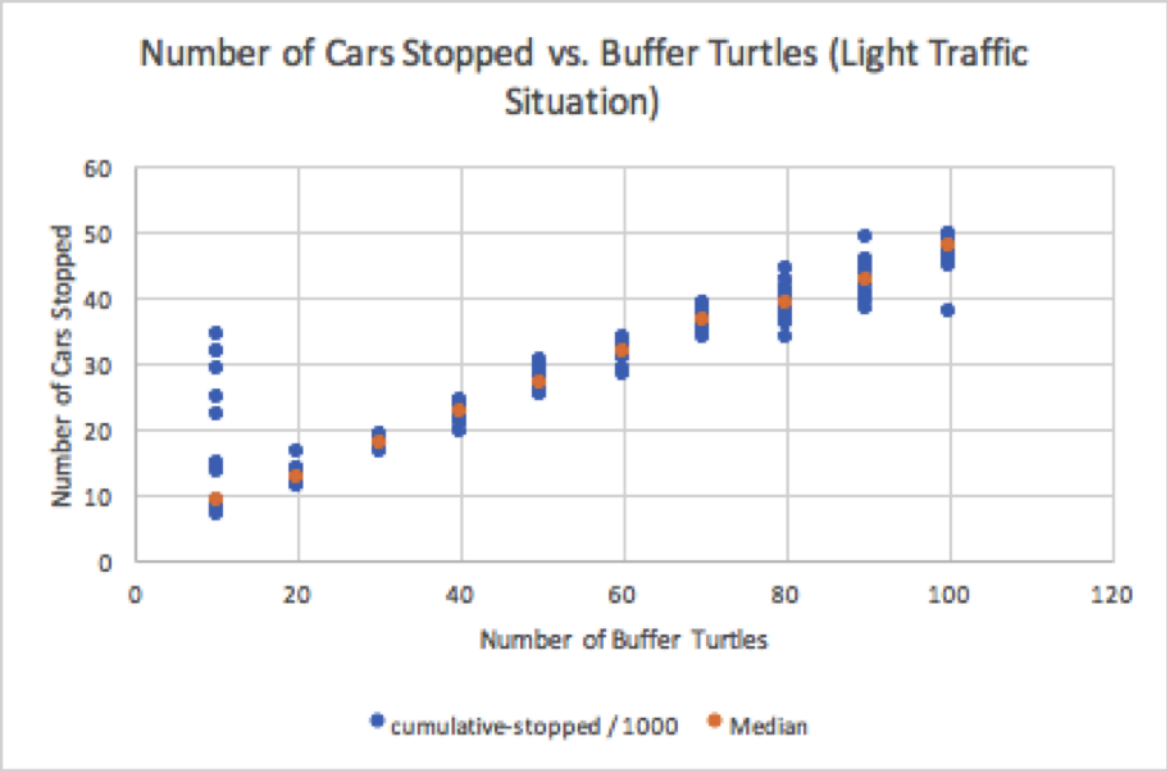

Figure 4: Number of stopped cars at different buffer levels.

An explanation of the graphs:

To test our model, we used BehaviorSpace. For each quantity of buffer turtles, the model went through 20 repetitions for average wait time and for the average number of cars stopped at any given tick. The time limit for each repetition was 1000 ticks=1000 seconds (about 17 minutes). Each blue dot on the graphs above represents one data point for each repetition at t=1000 for that quantity of buffer turtles. The median value for each quantity is in orange.

After examining a wide range of buffer turtle values, we found that the results varied based on our measure of success and on the amount of traffic. Regardless of the traffic situation, a larger buffer increased the average wait time. With a larger buffer, cars at a red light must wait longer for the traffic to pile up before they will get the green light; this will increase the average wait time.

However, the medians suggest that with a moderate buffer in a heavy traffic situation, fewer cars will be stopped. Based on the median number of cars stopped in a heavy traffic situation, 40 would be the ideal number of buffer cars. In a light traffic situation (see graph above), a lighter buffer is better by both measures. When the traffic is heavy only on one side, the minimum of the medians corresponds to 30 buffer turtles; this is a sensible conclusion since the ideal number for heavy traffic on both sides is 40 buffer turtles.

The result is that the model will have fewer cars waiting for longer; and if one turtle is unlucky enough to be the first one to a red light, it will probably be sitting there for a long time. In a light or one-sided traffic situation, a low buffer is better. In a heavy traffic situation, a buffer of around 40 cars will help reduce the number of stopped cars on average, but it will not decrease the average wait time. That's not to say that the buffer idea is all bad; a light buffer can keep the traffic light from switching all the time and only letting a few cars through.

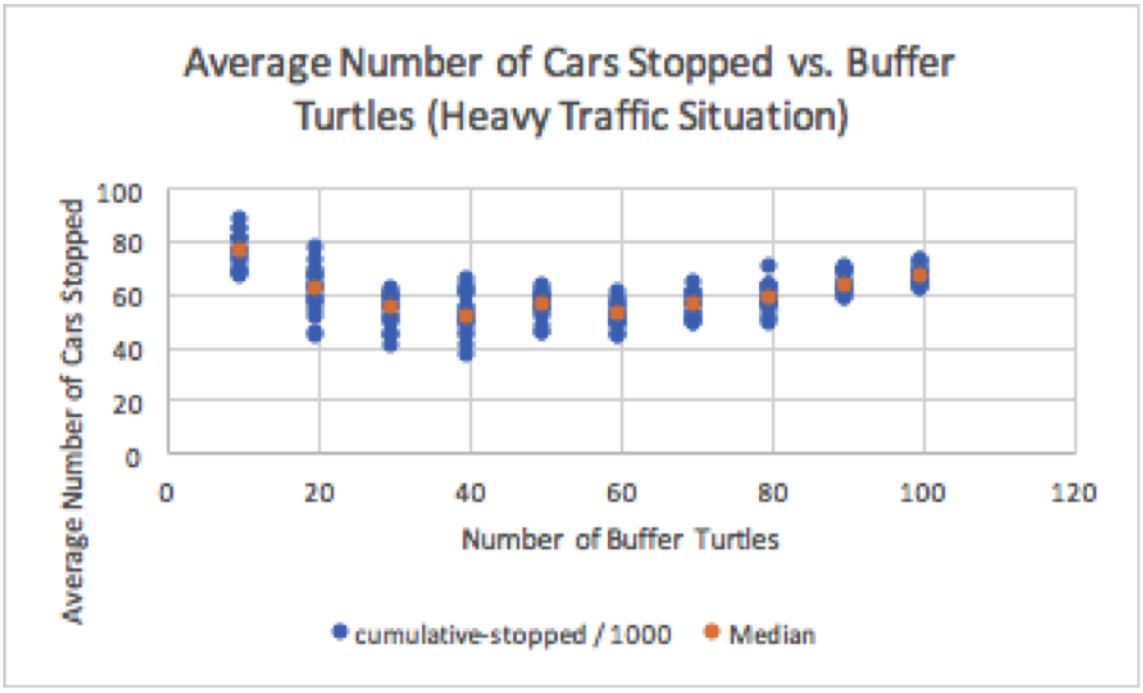

Figure 5: Average number of cars for heavy traffic scenario from both sides.

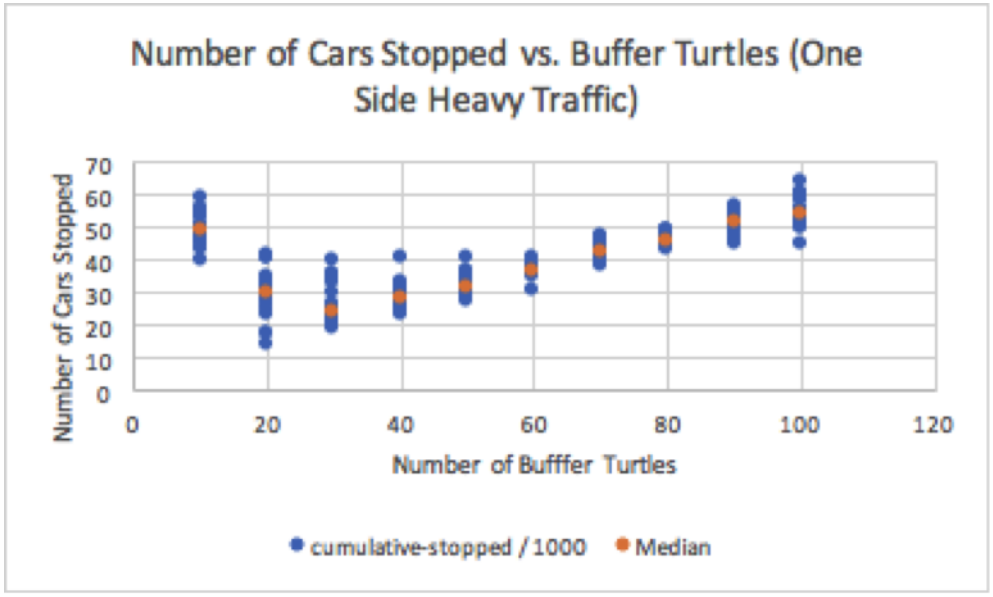

Figure 6: Average number of stopped cars for heavy traffic scenario from one side.

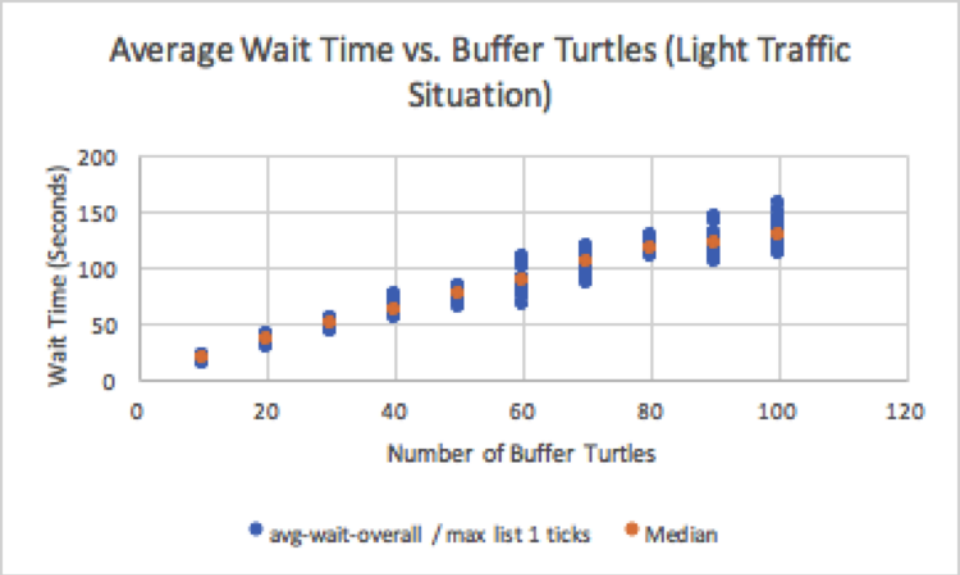

Figure 7: Average wait time for light Traffic scenario on both sides

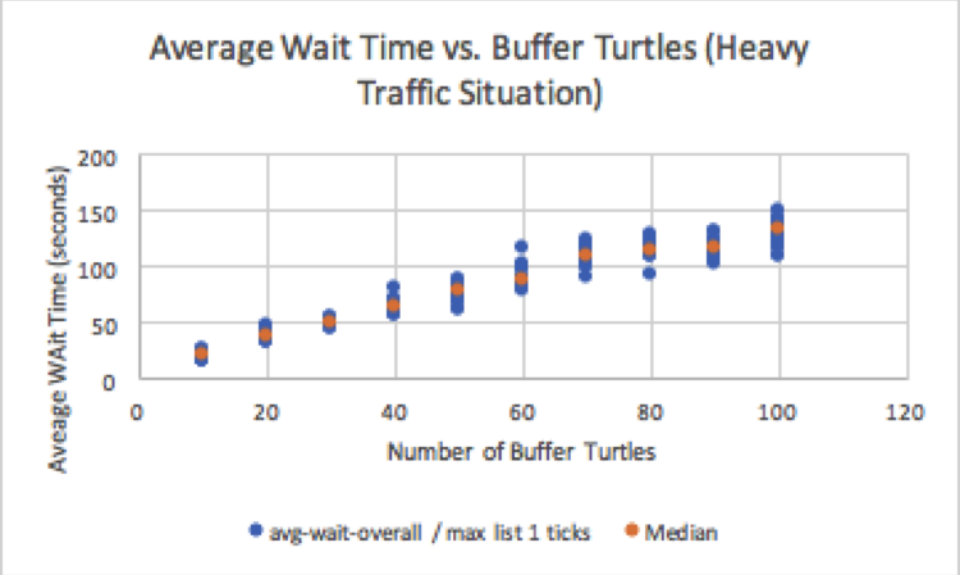

Figure 8: Average wait time for heavy Traffic scenario on both sides.

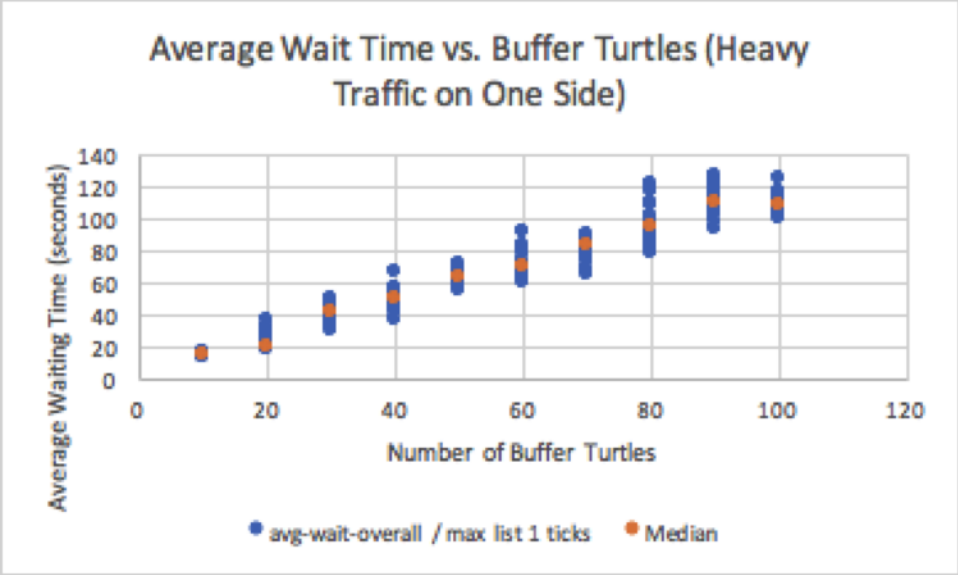

Figure 9: Average wait time for heavy Traffic scenario on one side

5. Conclusion

The model does give a good prediction of how this kind of traffic light would work, for better or for worse. In a light traffic situation, a buffer might not be necessary. In a situation with heavier traffic, a moderate buffer might lead to fewer cars stopped for longer (concentrating the misfortune on a few unlucky drivers!)

The model probably would not be realistic for an average traffic light. Drivers would probably feel it would be unfair and unjust to have to wait for other cars before they were allowed to go! Something like this might be used in a drawbridge setting, where perhaps the side with the most traffic is allowed to clear first. A situation like that might be a good real-life test to see how this model really works.

6. References

- Leise,T (2014). Modified NetLogo traffic model for Math 140.
- Wilensky, U. (1998). NetLogo Traffic Intersection model. http://ccl.northwestern.edu/netlogo/models/TrafficIntersection. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.
- Wilensky, U. (1999). NetLogo. http://ccl.northwestern.edu/netlogo/. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.